Homework 10 Solutions

Note: These solutions are not necessarily model answers. Rather, they are designed to be tutorial in nature, and sometimes contain a little more explanation than an ideal solution. Also, bear in mind that there may be more than one correct solution. The maximum total number of points available is 30.

1. (a) Since both X and μ are non-negative, we can write $|X-\mu| \le X+\mu$ and hence $\mathrm{E}[|X-\mu|] \le \mathrm{E}[X+\mu] = \mathrm{E}[X]+\mu = 2\mu$. *2pts*

(b) Consider the r.v. X that takes on the value 0 with probability $1-p$ and the value $M > \mu$ with probability p (where p, M are to be determined). The constraint that $\mathrm{E}[X] = \mu$ implies that $pM = \mu$. *3pts* Now we can compute

$$\mathrm{E}[|X-\mu|] = (1-p)\mu + p(M-\mu) = 2(1-p)\mu.$$

Thus by taking $p = \varepsilon/2$ (and $M = \mu/p = 2\mu/\varepsilon$), we get $\mathrm{E}[|X-\mu|] = (2-\varepsilon)\mu$ for any desired $\varepsilon > 0$, as required.

(c) In this case we can set let X take values $\mu + M$ and $\mu - M$ each with probability $\frac{1}{2}$, for an arbitrarily large M. Then $\mathrm{E}[|X-\mu|] = M$, which is unbounded. *1pt*

2. (a) We let $Z_i = (a_i, b_i)$, and let $L = L(Z_1, \ldots, Z_n)$ denote the length of a lcs of a, b. Then $X_i = \mathrm{E}[L|Z_1, \ldots, Z_i]$ is a martingale (the Doob martingale of L w.r.t. (Z_i)). It is easy to check that L is 2-Lipschitz (if we remove a_i, b_i and their partners from any common subsequence of a, b, we get a subsequence at most two shorter; and by reversing the argument we get a similar bound on the increase caused by changing a_i, b_i). Since the Z_i are also independent, we can apply Azuma's inequality with bounded differences of 2 to deduce that *4pts*

$$\Pr[|X_n - \mu_n| \ge \lambda] \le 2\exp(-\lambda^2/8n).$$

NOTE: We can actually do slightly better by considering instead the filter $Z_{2i-1} = a_i$, $Z_{2i} = b_i$, which makes L 1-Lipschitz with a difference sequence of length $2n$ and hence replaces the above bound by $2\exp(-\lambda^2/4n)$.

Note that independence of the Z_i is crucial here, in addition to the Lipschitz property; see part (ii) below for an illustration of what can go wrong in the absence of independence.

(b) Part (a) shows that deviations from the mean μ_n of order $\omega(\sqrt{n})$ (i.e., asymptotically larger than $\sqrt{n}$) are very unlikely. Since we are given that μ_n itself is linear in n, the concentration implied by part (a) is indeed useful as $\sqrt{n}$ is of lower order than n. *1pt*

(c) (i) No difference; the argument above is oblivious to the alphabet size. *1pt*

(ii) In the absence of independence, we can't claim any non-trivial concentration. (For example, suppose we have the following values for a, b, each with probability $\frac{1}{4}$: $(a = b = 0^n)$, $(a = b = 1^n)$, $(a = 0^n, b = 1^n)$ and $(a = 1^n, b = 0^n)$. Then $\mathrm{E}[L] = \frac{n}{2}$, but $|L - \mathrm{E}[L]| = \frac{n}{2}$ with probability 1.) *2pts*

Note that the arguments of part (a) do still work if a_i and b_i are dependent, provided that a_i, a_j are independent, and b_i, b_j are independent, for $i \ne j$.

(iii) Here the argument above still holds, but the function L becomes 3-Lipschitz. Thus we get the slightly weaker bound *2pts*

$$\Pr[|X_n - \mu_n| \geq \lambda] \leq 2\exp(-\lambda^2/18n).$$

NOTE: The alternative argument above also extends, making L 1-Lipschitz over a sequence of length $3n$ and giving the better bound $2\exp(-\lambda^2/6n)$.

3. We consider the Doob martingale $Y_i := \mathrm{E}[B_n \mid X_1, \ldots, X_i]$. Clearly the function B_n is 1-Lipschitz, since changing the size of one item can change the number of required bins by at most 1. Since the variables X_i are also independent, we may therefore use Azuma's inequality with all $c_i = 1$ to conclude that *4pts*

$$\Pr[|B_n - \mu_n| \geq \lambda] \leq 2\exp(-\lambda^2/2n).$$

Thus once again, deviations of size $\omega(\sqrt{n})$ from the mean are very unlikely, so the distribution is concentrated as claimed. The concentration result is oblivious to the distribution of the X_i (assuming of course that it is supported on the interval $(0, 1)$. (However, the value of $\mathrm{E}[B_n]$ does depend on the distribution.)

4. Throughout, we will interpret an $m \times m$ binary matrix Z as an m^2-bit binary number by writing out the matrix in row-major order (i.e., writing out the first row, then the second row, and so on), and we use as a fingerprint $F(Z) := Z \bmod p$, where p is a prime chosen at random from a suitable range. Writing $X[i, j]$ for the $m \times m$ submatrix of X with top left corner at position (i, j), we can apply the usual Karp-Rabin scheme as follows: *10pts*

```
for j = 1 to n − m + 1 do
  for i = 1 to n − m + 1 do
    if F(Y) = F(X[i, j]) then return "match"
return "no match"
```

Given $F(X[i, j])$ the next fingerprint $F(X[i + 1, j])$ can be computed quickly using the rule

$$F(X[i+1, j]) = \Big(2^m\big(F(X[i, j]) - 2^{m(m-1)}F(x[i, j])\big) + F(x(i + m, j))\Big) \bmod p,$$

where $x[i, j]$ denotes the m-bit number obtained from the matrix elements $x_{i,j}$ through $x_{i,j+m-1}$. If we have precomputed $F(x[i, j])$ for all i, j, this update takes time only $O(1)$, assuming that multiplication and addition mod p can be done in constant time. Each time the column is shifted right, however, the initial fingerprint $F(X[1, j])$ takes time $O(m)$ to compute, since each of the m rows must be handled separately. Therefore the total running time is $O(n^2 + nm) = O(n^2)$, plus the time to precompute the $F(x[i, j])$. This precomputation can also be handled column by column. The first column ($j = 1$) takes time $O(mn)$ since there are n rows to deal with, each one being an m-bit string. Subsequent columns can be handled more efficiently as follows:

$$F(x[i, j+1]) = \Big(2F(x[i, j]) - 2^m x_{i,j} + x_{i,j+m}\Big) \bmod p.$$

This computation takes only constant time, so that the remaining $F(x[i, j])$ (after the first column) can be computed in $O(n^2)$ time, giving a total precomputation time of $O(mn + n^2) = O(n^2)$ and a total running time of $O(n^2)$. For comparison, note that the naive algorithm, which explicitly compares Y against all submatrices $X[i, j]$, takes time $O(n^2m^2)$.

Just as in the one-dimensional case, if Y is contained in X then this algorithm is always correct. Otherwise, an error can occur only when p divides $\prod_{i,j} |Y - X[i, j]|$, which is an (m^2n^2)-bit number. Thus choosing p randomly from the primes in $\{2, \ldots, T\}$, where $T = cm^2n^2$ for a suitable modest constant c, yields a small probability of error. This means that p will have only $O(\log n)$ bits, so our assumption that arithmetic mod p can be done in constant time is justified.